


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

+(errors +AND +procedures +AND +calls) +AND +(memory +



THE ACM DIGITAL LIBRARY

[Feedback Report](#)

Terms used

errors AND procedures AND calls AND memory AND arguments AND procedures AND calls AND registers ,

Sort results by Display results
☒ [Save results to a Binder](#)

 Try an /
Try this

☒ [Search Tips](#)
☐ [Open results in a new window](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

1 [Lightweight remote procedure call](#)

Brian N. Bershad, Thomas E. Anderson, Edward D. Lazowska, Henry M. Levy

February 1990 **ACM Transactions on Computer Systems (TOCS)**, Volume 8 Issue 1

Full text available: pdf(1.60 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Lightweight Remote Procedure Call (LRPC) is a communication facility designed and optimized for on the same machine. In contemporary small-kernel operating systems, existing RPC systems include the type of communication that predominates—between protection domains on the same machine weakly related subsystems into the same protection domain, trading safety for ...

2 [Compiler transformations for high-performance computing](#)

David F. Bacon, Susan L. Graham, Oliver J. Sharp

December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4

Full text available: pdf(6.32 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

In the last three decades a large number of compiler transformations for optimizing programs have uniprocessors reduce the number of instructions executed by the program using transformations to data-flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel memory locality with transformations that rely on tracking the properties of ...

Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism,

3 [Data-Driven and Demand-Driven Computer Architecture](#)

Philip C. Treleaven, David R. Brownbridge, Richard P. Hopkins

January 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 1

Full text available: pdf(4.14 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

4 [The family of concurrent logic programming languages](#)

Ehud Shapiro

September 1989 **ACM Computing Surveys (CSUR)**, Volume 21 Issue 3

Full text available: pdf(9.62 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Concurrent logic languages are high-level programming languages for parallel and distributed systems and novel concurrent programming techniques. Being logic programming languages, they preserve the programming model, including the logical reading of programs and computations, the convenience of terms and manipulating them using unification, and the amenability to metaprogramming ...


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

(extracting information call linkage) AND (optimizing procedure calls)

131,821

THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used

extracting information call linkage AND optimizing procedure calls

Found 76,820 of 132,857

Sort results by

relevance

[Save results to a Binder](#)
[Try an Advanced Search](#)

Display results

expanded form

[Search Tips](#)
[Try this search in The ACM Guide](#)
☐ Open results in a new window

Results 1 - 20 of 200

 Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale ☐ ☐ ☐ ☐ ☐

1 [The impact of interprocedural analysis and optimization in the Rⁿ programming environment](#)

Keith D. Cooper, Ken Kennedy, Linda Torczon

 August 1986 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 8 Issue 4

 Full text available: [pdf\(2.87 MB\)](#)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

In spite of substantial progress in the theory of interprocedural data flow analysis, few practical compiling systems can afford to apply it to produce more efficient object programs. To perform interprocedural analysis, a compiler needs not only the source code of the module being compiled, but also information about the side effects of every procedure in the program containing that module, even separately compiled procedures. In a conventional batch compiler system, the increase in compil ...

2 [The impact of interprocedural analysis and optimization on the design of a software development environment](#)

Keith D. Cooper, Ken Kennedy, Linda Torczon

 June 1983 **Proceedings of the ACM SIGPLAN 85 symposium on Language issues in programming environments**, Volume 18, 20 Issue 6, 7

 Full text available: [pdf\(971.84 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

One of the primary goals of the IRⁿ programming environment project is to mount a concerted attack on the problems of performing interprocedural analysis and optimization in a compiler. Few commercial optimizing compilers employ interprocedural techniques because the cost of gathering the requisite information in a traditional compiler is too great. Computing the side effects of a procedure call requires detailed knowledge of the internals of both the called procedure a ...

3 [Fast object-oriented procedure calls: lessons from the Intel 432](#)

E. F. Gehringer, R. P. Colwell

 June 1986 **ACM SIGARCH Computer Architecture News, Proceedings of the 13th annual international symposium on Computer architecture**, Volume 14 Issue 2

 Full text available: [pdf\(995.24 KB\)](#)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

As modular programming grows in importance, the efficiency of procedure calls assumes an ever more critical role in system performance. Meanwhile, software designers are becoming more aware of the benefits of object-oriented programming in structuring large software systems. But object-oriented programming requires a good deal of support, which can best be distributed between the compiler and architectural levels. A major part of this support relates to the execution of procedure calls. Mus ...